IJDAR (2011) 14:229–239
DOI 10.1007/s10032-010-0135-3

ORIGINAL PAPER

Digital weight watching: reconstruction of scanned documents

Maarten Marx · Tim Gielissen

Received: 7 December 2009 / Revised: 2 July 2010 / Accepted: 14 October 2010 / Published online: 31 October 2010

Abstract A web portal providing access to over 250.000 scanned and OCRed cultural heritage documents is analyzed. The collection consists of the complete Dutch Hansard from 1917 to 1995. Each document consists of facsimile images of the original pages plus hidden OCRed text. The inclusion of images yields large file sizes of which less than 2% is the actual text. The search user interface of the portal provides poor ranking and not very informative document summaries (snippets). Thus, users are instrumental in weeding out non-relevant results. For that, they must assess the complete documents. This is a time-consuming and frustrating process because of long download and processing times of the large files. Instead of using the scanned images for relevance assessment, we propose to use a reconstruction of the original document from a purely semantic representation. Evaluation on the Dutch dataset shows that these reconstructions become two orders of magnitude smaller and still resemble the original to a high degree. In addition, they are easier to speed-read and evaluate for relevance, due to added hyperlinks and a presentation optimized for reading from a terminal. We describe the reconstruction process and evaluate the costs, the benefits, and the quality.

Keywords XML · Information extraction · Scanned documents

M. Marx (✉) · T. Gielissen
ISLA, University of Amsterdam, Science Park 107,
1098 XG Amsterdam, The Netherlands
e-mail: maartenmarx@uva.nl

1 Introduction

This paper addresses the fact that scanned and OCRed documents tend to be very large in file size because of the inclusion of facsimile[1] images. This makes them expensive to store, expensive to serve over the internet, and cumbersome to handle by users.

We present a case study in which we extracted the content and the structure from a large digitized corpus and reconstructed the documents from scratch "as new". This yielded much smaller (less than 1% of the original size when stored in gzipped XML, 1.5% when stored as PDF) and far better readable documents which in addition are easier to browse because of added hyperlinked structure.

We present the data, describe our techniques, evaluate the results, and generalize them to other cases.

The novelty of our work is located in the way we reconstruct the original files. The reconstruction is based on a purely semantical representation of the original data. The reconstruction is done using only two well-described declarative programming languages: XSLT 2.0 and LaTeX.

Outline The rest of this introduction motivates our research problem. Section 2 describes the data; Sect. 3 analyzes the current search user interface. Section 4 contains the two transformation processes: structure extraction and document reconstruction. We evaluate the quality of these two transformations in Sect. 5. Section 6 contains related work. We conclude and generalize our results in Sect. 7.

Motivation Our work is motivated by an analysis from a user perspective of a web portal providing access to over 250.000 scanned and OCRed cultural heritage documents. The collection consists of the complete Dutch Hansard from

[1] By a facsimile image, we mean a document that visually resembles the original.

Table 1 Description of the corpus used for experiments: proceedings of plenary meetings of the Dutch Parliament (both Houses), from 1980 to 1985

year	Number of documents	Total file size (GB)	Number of words	Number of pages
1980	143	2.52	7,943,904	7.709
1981	124	1.76	5,613,432	5.405
1982	133	1.76	5,552,685	5.544
1983	150	2.50	7,714,903	7.612
1984	147	2.54	7,870,318	7.750
1985	146	2.56	8,251,097	8.081
Total	843	13.62	42,946,339	42.101

1917 to 1995. Each document consists of facsimile images of the original pages plus hidden OCRed text. The inclusion of images for each page yields large file sizes. The average document in our collection contains 51 thousand words, is 50 pages long and, has a file size of 16.5 MB.

Documents consisting of facsimile images become a problem when the search engine for those documents returns relevance-ranked lists with low precision. In this case, users are instrumental in weeding out non-relevant results. Users have two pieces of information for this task [10]: the document summary given in the list of results (the "snippet") and the document itself. If the snippet does not give enough information to reject the result, the user has to download and inspect the actual document.

One of the most pressing problems arising from large file sizes are long download times. For instance, with a fast (12 Mbit/s) internet connection, it takes in the optimal case 10 s to download an average document (16 MB) from our collection. In reality, this speed is often up to 20 times slower. Such long download times prohibit natural work flows.

Inspection of a document consisting of facsimile images is also hindered by large sizes. In addition, facsimiles with hidden OCRed text are difficult to quick scan because they are often hard to read and there is no support for navigation in the document.

These long waiting and inspection times become a source of frustration when the downloaded document was ranked high by the search engine but found out to be not relevant. An obvious solution to this problem is to use a search engine with better ranking and excellent document summaries. In reality, this option is often not feasible.

To solve this problem, we propose to add an intermediate document summary between the snippet and the document, very much like the "Quick Look" button in Apple's Mail program. It consists of the complete OCRed text specially presented for fast scanning by humans. It is an approximation of the original text because of the OCR errors and differences in layout. Only when the document turns out to be relevant, the large file with all facsimile images needs to be downloaded.

2 Description of the data

The Netherlands have parliamentary proceedings since 1814. From 1995, these are available as digitally produced PDF files. The Dutch Royal Library together with the Dutch parliament has scanned and OCRed all proceedings from 1814 until 1995. The collection consists of 2.5 million pages which physically span 150 m. The digital copy needs approximately 30 Terra Byte storage space.

The proceedings are available online at

http://www.statengeneraaldigitaal.nl.

Each document is a combination of files: metadata in XML, a JPEG image for each page, the OCRed text in an XML wrapper, and an MPEG21-DIDL file describing the connections between all these files [17]. Each document is also available as a PDF file that combines all this information.

In November 2009, all documents from 1917 to 1995 are available. The complete corpus is planned to be online in the fall of 2010.

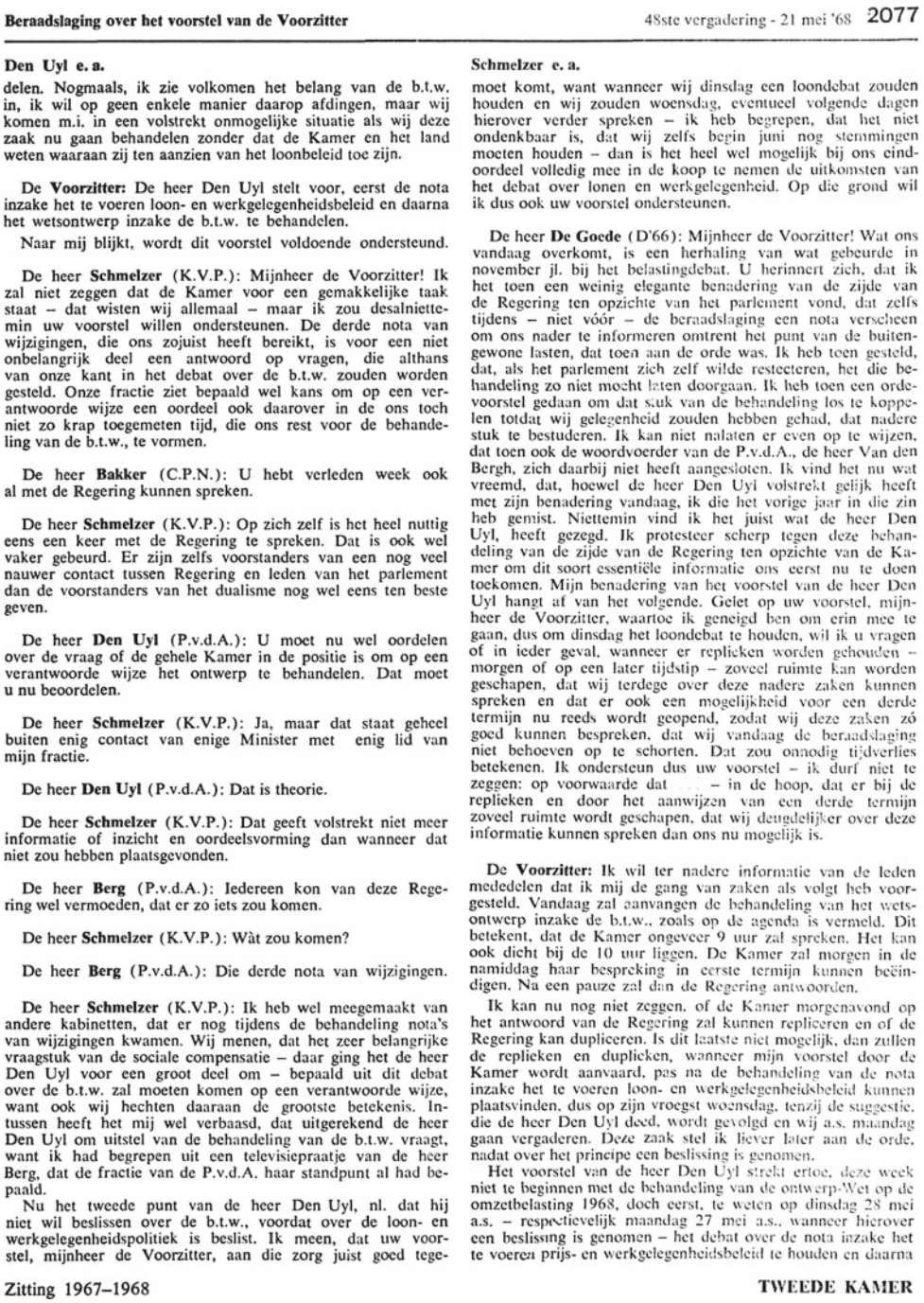
We will refer to the part of the collection that consists of verbatim notes of plenary meetings of the House of Parliament as the *SGD* corpus. The SGD corpus consists of 12.796 documents, covering 383.863 pages. It has over 100 million tokens of which 204 thousand occur more than once and 55 thousand occur at least 20 times [22].

Our experiments are based on the documents from 1980 to 1985. Each document contains the meeting notes of 1 day. Table 1 describes the yearly production.

As stated before, on average, one document contains 51 thousand words, is 50 pages long, and has a file size of 16.5 MB. Because each document represents the meeting notes of one complete day, on an average day in Dutch parliament 51 thousand words are recorded.

The largest document is 49 MB (151 thousand words), and the smallest is just 1 page, with 382 words. At the meeting of this one page document, less than half of the Dutch MP's were present and then by law the meeting cannot start.

Figure 1 shows the facsimile of a typical page. The layout is rather simple: a header and a footer and a body of text set in two columns. The text is divided into blocks according to



Beraadslaging over het voorstel van de Voorzitter 48ste vergadering - 21 mei '68 2077

Den Uyl e. a.

delen. Nogmaals, ik zie volkomen het belang van de b.t.w. in, ik wil op geen enkele manier daarop afdingen, maar wij komen m.i. in een volstrekt onmogelijke situatie als wij deze zaak nu gaan behandelen zonder dat de Kamer en het land weten waaraan zij ten aanzien van het loonbeleid toe zijn.

De **Voorzitter:** De heer Den Uyl stelt voor, eerst de nota inzake het te voeren loon- en werkgelegenheidsbeleid en daarna het wetsontwerp inzake de b.t.w. te behandelen.

Naar mij blijkt, wordt dit voorstel voldoende ondersteund.

De heer **Schmelzer** (K.V.P.): Mijnheer de Voorzitter! Ik zal niet zeggen dat de Kamer voor een gemakkelijke taak staat – dat wisten wij allemaal – maar ik zou desalniettemin uw voorstel willen ondersteunen. De derde nota van wijzigingen, die ons zojuist heeft bereikt, is voor een niet onbelangrijk deel een antwoord op vragen, die althans van onze kant in het debat over de b.t.w. zouden worden gesteld. Onze fractie ziet bepaald wel kans om op een verantwoorde wijze een oordeel ook daarover in de ons toch niet zo krap toegemeten tijd, die ons rest voor de behandeling van de b.t.w., te vormen.

De heer **Bakker** (C.P.N.): U hebt verleden week ook al met de Regering kunnen spreken.

De heer **Schmelzer** (K.V.P.): Op zich zelf is het heel nuttig eens een keer met de Regering te spreken. Dat is ook wel vaker gebeurd. Er zijn zelfs voorstanders van een nog veel nauwer contact tussen Regering en leden van het parlement dan de voorstanders van het dualisme nog wel eens ten beste geven.

De heer **Den Uyl** (P.v.d.A.): U moet nu wel oordelen over de vraag of de gehele Kamer in de positie is om op een verantwoorde wijze het ontwerp te behandelen. Dat moet u nu beoordelen.

De heer **Schmelzer** (K.V.P.): Ja, maar dat staat geheel buiten enig contact van enige Minister met enig lid van mijn fractie.

De heer **Den Uyl** (P.v.d.A.): Dat is theorie.

De heer **Schmelzer** (K.V.P.): Dat geeft volstrekt niet meer informatie of inzicht en oordeelsvorming dan wanneer dat niet zou hebben plaatsgevonden.

De heer **Berg** (P.v.d.A.): Iedereen kon van deze Regering wel vermoeden, dat er zo iets zou komen.

De heer **Schmelzer** (K.V.P.): Wàt zou komen?

De heer **Berg** (P.v.d.A.): Die derde nota van wijzigingen.

De heer **Schmelzer** (K.V.P.): Ik heb wel meegemaakt van andere kabinetten, dat er nog tijdens de behandeling nota's van wijzigingen kwamen. Wij menen, dat het zeer belangrijke vraagstuk van de sociale compensatie – daar ging het de heer Den Uyl voor een groot deel om – bepaald uit dit debat over de b.t.w. zal moeten komen op een verantwoorde wijze, want ook wij hechten daaraan de grootste betekenis. Intussen heeft het mij wel verbaasd, dat uitgerekend de heer Den Uyl om uitstel van de behandeling van de b.t.w. vraagt, want ik had begrepen uit een televisiepraatje van de heer Berg, dat de fractie van de P.v.d.A. haar standpunt al had bepaald.

Nu het tweede punt van de heer Den Uyl, nl. dat hij niet wil beslissen over de b.t.w., voordat over de loon- en werkgelegenheidspolitiek is beslist. Ik meen, dat uw voorstel, mijnheer de Voorzitter, aan die zorg juist goed tege-

Schmelzer e. a.

moet komt, want wanneer wij dinsdag een loondebat zouden houden en wij zouden woensdag, eventueel volgende dagen hierover verder spreken – ik heb begrepen, dat het niet ondenkbaar is, dat wij zelfs begin juni nog stemmingen moeten houden – dan is het heel wel mogelijk bij ons eindoordeel volledig mee in de koop te nemen de uitkomsten van het debat over lonen en werkgelegenheid. Op die grond wil ik dus ook uw voorstel ondersteunen.

De heer **De Goede** (D'66): Mijnheer de Voorzitter! Wat ons vandaag overkomt, is een herhaling van wat gebeurde in november jl. bij het belastingdebat. U herinnert zich, dat ik het toen een weinig elegante benadering van de zijde van de Regering ten opzichte van het parlement vond, dat zelfs tijdens – niet vóór – de beraadslaging een nota verscheen om ons nader te informeren omtrent het punt van de buitengewone lasten, dat toen aan de orde was. Ik heb toen gesteld, dat, als het parlement zich zelf wilde restecteren, het die behandeling zo niet mocht laten doorgaan. Ik heb toen een ordevoorstel gedaan om dat stuk van de behandeling los te koppelen totdat wij gelegenheid zouden hebben gehad, dat nadere stuk te bestuderen. Ik kan niet nalaten er even op te wijzen, dat toen ook de woordvoerder van de P.v.d.A., de heer Van den Bergh, zich daarbij niet heeft aangesloten. Ik vind het nu wat vreemd, dat, hoewel de heer Den Uyl volstrekt gelijk heeft met zijn benadering vandaag, ik die het vorige jaar in die zin heb gemist. Niettemin vind ik het juist wat de heer Den Uyl, heeft gezegd. Ik protesteer scherp tegen deze behandeling van de zijde van de Regering ten opzichte van de Kamer om dit soort essentiële informatie ons eerst nu te doen toekomen. Mijn benadering van het voorstel van de heer Den Uyl hangt af van het volgende. Gelet op uw voorstel, mijnheer de Voorzitter, waartoe ik geneigd ben om erin mee te gaan, dus om dinsdag het loondebat te houden, wil ik u vragen of in ieder geval, wanneer er replicken worden gehouden – morgen of op een later tijdstip – zoveel ruimte kan worden geschapen, dat wij terdege over deze nadere zaken kunnen spreken en dat er ook een mogelijkheid voor een derde termijn nu reeds wordt geopend, zodat wij deze zaken zó goed kunnen bespreken, dat wij vandaag de beraadslaging niet behoeven op te schorten. Dat zou onnodig tijdverlies betekenen. Ik ondersteun dus uw voorstel – ik durf niet te zeggen: op voorwaarde dat – in de hoop, dat er bij de replieken en door het aanwijzen van een derde termijn zoveel ruimte wordt geschapen, dat wij deugdelijker over deze informatie kunnen spreken dan ons nu mogelijk is.

De **Voorzitter:** Ik wil ter nadere informatie van de leden mededelen dat ik mij de gang van zaken als volgt heb voorgesteld. Vandaag zal aanvangen de behandeling van het wetsontwerp inzake de b.t.w., zoals op de agenda is vermeld. Dit betekent, dat de Kamer ongeveer 9 uur zal spreken. Het kan ook dicht bij de 10 uur liggen. De Kamer zal morgen in de namiddag haar bespreking in eerste termijn kunnen beëindigen. Na een pauze zal dan de Regering antwoorden.

Ik kan nu nog niet zeggen, of de Kamer morgenavond op het antwoord van de Regering zal kunnen repliceren en of de Regering kan dupliceren. Is dit laatste niet mogelijk, dan zullen de replieken en duplicken, wanneer mijn voorstel door de Kamer wordt aanvaard, pas na de behandeling van de nota inzake het te voeren loon- en werkgelegenheidsbeleid kunnen plaatsvinden, dus op zijn vroegst woensdag, tenzij de suggestie, die de heer Den Uyl deed, wordt gevolgd en wij a.s. maandag gaan vergaderen. Deze zaak stel ik liever later aan de orde, nadat over het principe een beslissing is genomen.

Het voorstel van de heer Den Uyl strekt ertoe, deze week niet te beginnen met de behandeling van de ontwerp-Wet op de omzetbelasting 1968, doch eerst, te weten op dinsdag 28 mei a.s. – respectievelijk maandag 27 mei a.s., wanneer hierover een beslissing is genomen – het debat over de nota inzake het te voeren prijs- en werkgelegenheidsbeleid te houden en daarna

Zitting 1967–1968 TWEEDE KAMER

Fig. 1 A typical page of the Dutch parliamentary proceedings. Page 2077 of the meeting of May 21, 1968. Available at http://resourcessgd.kb.nl/SGD/19671968/PDF/SGD_19671968_0000410.pdf (22 MB)

whom is speaking. Each block begins with the name of the speaker in boldface.

3 Analysis of the search interface

We provide an analysis of the search user interface at http://www.statengeneraaldigitaal.nl, the portal that serves the Dutch parliamentary proceedings. The ranking of the results after a keyword search is rather poor. Thus, the user is instrumental in weeding out non-relevant documents. We analyse the tools provided by the search user interface which assist in this task. Based on the recommendations in [10], we give a list of positive and negative aspects of the interface.

The search interface has a standard three-layer architecture: a search page that leads to a result page (often called

"SERP") with "ten blue links" which lead to the actual documents. We discuss each layer.

The search page has a standard and an advanced interface. The advanced search page facilitates the formulation of precise queries by offering selections on three natural facets for this collection. One can select the House, make a date restriction, and choose among three document types (meeting notes, written questions and answers, and documents sent to the Parliament). These three facets come back in various parts of the search interface. For each document type, specific further restrictions are possible.

The result page shows the first ten hits with the option to view the next ten results, as usual. Besides the ten results, there is aggregation information and faceted search machinery [9]. The page shows the total number of hits and the distribution of the hits over the values of the three mentioned facets: document type (3 values), House (4 values), and parliamentary year (at the time of writing over 70 values). Clicking on a facet value combination restricts the search, and the distribution of hits over the facets is recomputed.

Hearst [10] calls each hit a "document surrogate" in order to highlight their function: help the user to understand the primary object. In particular:

> The quality of the document surrogate has a strong effect on the ability of the searcher to judge the relevance of the document.

The document surrogates of SGD consist of three pieces of information:

- A title with a hyperlink to the document. The title provides the values on the three facets plus some additional information (e.g. the date in case of proceedings). It is query independent.
- The number of pages of the document.
- An extract from the retrieved document ("snippet"). The snippet consists of pieces of text taken from various parts of the document which are concatenated. These snippets appear mostly query independent.
  Query terms are not highlighted in the snippets.

On a sample of 617 snippets (all taken from the first result page after a query), less than one-third contained the query term. The snippets are rather long: on average 383 characters ($N=612$). This is more than twice the length of the snippets at Google.

We now discuss the third layer, the actual documents. SGD is a vertical search engine that only serves documents under its control, and SGD can thus determine their presentation. Each document is shown in a special viewer which implements an entry point retrieval [30] system: the user is brought to the first page in the document in which the search term occurs. From there, the user can go backward and forward through the pages and jump to following and preceding entry points. The user views a JPEG image of the page on which the search term is highlighted. It is also possible to view the OCRed text of the page as an HTML document.

3.1 Evaluation

We evaluate the three layers of the search system with respect to the ease and speed in which the following task can be performed:

> From the list of retrieved documents, find those that are relevant.

We first list aspects that have a positive effect on the task, followed by those that have a negative effect. We conclude with proposals for improvement. These will be further developed and evaluated in the rest of the paper.

Positive features

- (++) The advanced search interface makes stating precise queries possible and easy. This is especially useful when the user has already good knowledge about the desired document(s) (as in known-item search).
- (+) The faceted search machinery is useful for browsing the collection and quickly zooming in on parts of the results. The temporal facet suffers from a large amount of values, overcrowding the page and resulting in flat frequency distributions.
  Following [9], a hierarchical faceted design is to be preferred. One level above parliamentary years are the legislative periods (in The Netherlands maximally 4 years). These can be grouped into political eras. For the period 1918–1995, this would yield six eras each having between 2 and 8 legislative periods.[2]
- (++) Entry point retrieval with search term highlighting provides direct access to the potentially relevant parts in the often very long documents. With frequent words in long documents, it would be desirable to have a ranking of relevant pages within one document as well [30,21].

Negative features

- (−−) There is no apparent ranking of the search results. It is not possible for users to specify orderings or groupings on metadata.
- (−−) The document summaries are mostly query independent. This makes them basically useless for relevance assessment of the underlying document [4]. That paper

[2] http://www.parlement.com/: Kabinetten per tijdvak.

also suggests that snippets should pass a simple readability test. The long SGD snippets consisting of sentences from different parts of the document are often unintelligible.
– (− − −) Retrieving a document takes a long (download) time. Browsing through the document takes a long (download) time. Reading the scanned images is difficult and hence time-consuming because of small font size, unfamiliar fonts, poor quality of the scan, and layout designed for paper printing. See Fig. 1.

Possible improvements The improvements we suggest are based on the premise that we cannot change the ranking by relevance. Thus, the user still has to do most of the relevance assessments. The above discussion yields four clear cases for improvement:

1. Improve aggregated search results and facets [24].
2. Offer (reverse) chronological ranking, possibly combined with result grouping [10].
3. Offer query-dependent document summaries with keyword highlighting [4].
4. Drastically improve download times and fast browsability of documents.

Of these improvements only the last is specific for noisy-data collections. The only way to drastically reduce download times is to postpone serving large images as long as possible.

3.2 Extended document summaries

We propose to add a fourth layer to the search architecture. It is placed in between the result page with the ten snippets and the actual documents. This functionality is comparable to the "Quick Look" button in Apple's Mail program. The fourth layer contains an approximation of the original document, based on the OCRed text and the original layout. Its sole purpose is to provide a fast interface to the complete document in order to make a quick relevance assessment.

We discuss and evaluate three systems for creating these document summaries. They differ in the amount of semantics that is explicit in the markup.

1. The simplest transformation preserves physical layout using absolute positions of each line of text and font information. This can be achieved efficiently and effectively with the `pdftohtml` package.[3] The only structural element that is preserved is the page. The size of resulting documents is on average 9% of the original size.

[3] http://pdftohtml.sourceforge.net/.

Table 2 Evaluation of the quality of the transformation from PDF to XML described in [22]

Semantic feature	Accuracy (%)
Page header	100
Page footer	89
Reading order	67
Paragraphs	91

Accuracy measures the number of times a feature is correctly extracted. For reading order, this means that the XML document order of the text on one complete page is the reading order of a the PDF page (usually in multiple columns), except for special text blobs like page headers, footers, and captions. Measured on 1034 pages from 15 PDF documents

2. Marx and Schuth [22] describes a system for transforming PDF files into an XML format in which page, paragraph, page number, page header, and page footer information is preserved. The program also attempts to detect the reading order of the multicolumn input and outputs the text in reading order. We evaluated its effectiveness on 1034 pages from 15 randomly chosen proceedings files from the period 1928–1966. The results are in Table 2. The size of resulting documents is on average 6% of the original size. The difference in compression with the previous transformation is due to the fact that here position information is only retained for each paragraph, not for each line of text.
3. The system described in this paper first does an extensive text analysis and produces semantically rich XML without any layout information except for paging. From that XML file, a uniform looking PDF file according to the style of one specific period (the 80ties) is created. The size of these PDF's is 1.5% of the original size.

This last system is described in the next section.

4 Description of the transformation

The transformation of a scanned and OCRed PDF document into a PDF document that closely resembles the original, but without the facsimile images, involves two steps. First, the structure that is implicit in the document is made explicit using a variety of text extraction techniques [6]. Next, the resulting XML document is transformed into a PDF document without facsimile images using XSLT and LaTeX. In this section, we describe both steps.

4.1 Making structure explicit: from PDF to XML

All documents contain structure. Often, we can easily recognize titles, paragraphs, and page numbers on the basis of their layout (e.g., position or size) and their content

(e.g., a number). This structure is in most cases not explicit in digital form, especially not when the document is scanned and OCRed. If the structure of documents in a corpus is standardized to some degree, the structure can be made explicit in digital form automatically [5].

The Dutch parliamentary proceedings show this kind of standardization of document structure. All elements of a proceeding, for example topics and speakers, are represented in their own distinct way. This enables the automatic recognition of these elements. We implemented the text and structure extraction as an Extract-Transfer-Load process [27] consisting of eight steps.

First, we extract the text from the PDF using the open-source program `pdftohtml` with the `-xml` option. This yields an XML file with for each line of text four coordinates that indicate the bounding box of that text. Multiple columns are detected and preserved. Some font and layout information is preserved but not all. The XML structure is simple and flat:

```
root ⟶ (page)*
page ⟶ (text)*
text ⟶ (#PCDATA,b,i,span)*
```

The second step involves cleaning the output from `pdftohtml`. We ensure that the output is well-formed XML, and we solve problems with diacritics. In this step, we also fix the most common OCR errors in this collection. We found one specific error quite often: the OCR inserts a space before the last letter of a word. For example, the token **wij** is OCRed as **wi j**. A regular expression designed to fix this problem matches 3.1% of all the lines in the text corpus (i.e., one line in one column in the original PDF as in Fig. 1). A sample of random 100 matches indicated that this procedure has a precision of 93%.

The values indicating the position and size of the bounding box of the text are normalized in the third step because they can differ among different devices.

In the fourth step, we analyze the document's layout. The margins, the number of columns, and the header and footer are detected and marked. For this, we use the position of the text elements and the (deducted) position of the whitespace. Using this information, we sort the text of the body (i.e., not belonging to the header or the footer) in reading order in the fifth step.

During the sixth and seventh step, different markers are placed in the text to signal the start of different elements in the document. In the sixth step, we place markers indicating the position of text elements in the document. We place markers on places where paragraphs begin (they are indented), where there is whitespace, and when a new column starts. This information is used in the seventh step where markers are placed based on the content of the document. In the seventh step, we use regular expressions to recognize standardized structure. We indicate this by an example. The start of a statement by a new person is signaled in the text by the title, the name, and the party of the speaker, as in

Mevrouw **Swenker** (VVD):

This adheres to the following structure: title, last name, and party name within parenthesis, colon. This is then followed by the statement this person made. The start of a statement can only begin after a whiteline marker or the start of a new column. So in our XML, we convert this to:

```
<speechstart speaker='Swenker' party='VVD' …/>,
```

with the … containing additional information.

We now have an XML document that is flat and contains markers that mark the beginning of structural elements, but not the end. In the last step, we replace the markers by XML tags that enclose the entire element. This is done by performing a cascade of groupings starting with the elements which need to be most deeply nested: the paragraphs `p`. XSLT 2.0 has a useful command for this task: `xsl:for-each-group`.

The result is an XML file with the same text as the original document but with explicit structure. The file is valid with respect to a rich Relax NG schema, constraining both the structure of the XML tree as well as the values of many attributes. The structure can be used for many purposes, e.g., to analyse the structure of the debates [12].

4.2 Reconstruction of the originals: from XML to PDF

Because the structural elements of the documents are made explicit in the XML file, it is possible to reconstruct the original PDF with high resemblance. We use a combination of XSLT 2.0 and LaTeX for this process. We created a XSLT stylesheet that transforms the XML file described in the previous subsection into a LaTeX file. We briefly describe this transformation.

First, the stylesheet writes a general preamble that loads all necessary packages and specifies layout information for the document. Some values are copied from the XML file, like the value that indicates the number of columns.

After the preamble, the document itself begins. For every element specified in the Relax NG schema, we defined a template (for more information about the schema, see [6]). These templates are nested according to the structure of the proceeding. First, we have a template for the topics, the highest level of the structure of the proceedings. The stylesheet writes the necessary layout information for the topic and then places the text from the XML file in the LaTeX file. Next, it applies all the templates for the elements within the topic. This way, we cycle through all the elements in the deep

Table 3 Total file sizes of the test corpus described in Table 1

	Size in MB	% of original
Original corpus	13.620	
Reconstructed PDF	205	1.5
gzipped XML	88	0.6

Table 4 Processing speeds of the two main transformations

PDF-2-XML	XML-2-PDF
27.6 pages/ min	1,503.6 pages/ min

structure of the XML and create the LaTeX file step-by-step. If the stylesheet encounters a pagebreak, it redefines the page style (including header and footer) for the next page.

When the LaTeX file is created, the next step is to create the PDF document with `pdflatex`. Creating the LaTeX file with XSLT and compiling the PDF file take about 1–2 s, depending on the size of the file.

An alternative approach for reconstructing the files is to use XSL-FO to produce a PDF document directly with XSLT. We used LaTeX because it appeared to be easier to obtain fast results.

5 Evaluation

We present a technical, an information-theoretic and an economic evaluation. The technical evaluation describes the reduction in file sizes obtained and the processing times needed for the reduction. In the economic evaluation, we compare the efforts invested in creating the transformation scripts with the benefits of smaller files and explicit markup information. In the information-theoretic evaluation, we look at the quality of the transformations: we evaluate whether information was lost or distorted.

5.1 Benefits of the reconstruction

Size reduction Our first goal was to reduce the file sizes. This was achieved with a reduction of 2 orders of magnitude. Table 3 lists the results.

Processing times Transforming the whole corpus of 13.62 GB into reconstructed PDF's took 25 h and 50 min, an average of 1.8 min per document. These times were measured on an Apple MacBook with 2.4 GHz CPU running the OS X 10.5.6 operating system.

The table below shows the percentage of the total time that was spent on each of the elements of the entire pipeline from Sect. 4.

pdftohtml	**PDF-2-XML** (without pdftohtml)	**XML-2-PDF**
40%	58%	2%

Table 4 shows the processing speeds of the two main steps of the transformation in pages per minute. (Recall that a typical document has about 50 pages.)

Both transformations can be done offline, and we need only to store the reconstructed PDF on the web server. The transformation from XML to PDF is fast enough to perform at query time. Recall that an average document is 50 pages. Thus, this can be transformed from a gzipped XML file into PDF in 2 s. This transformation can be further optimized as a streaming transformation [18].

Digital sustainability We aimed to make our transformations in such a way that they can be performed again after minimally 100 years with relative ease. Thus, we wanted a minimal dependency on specific software and hardware and transparent and reproducible transformations [7]. Our goal was to have all steps of the transformation written in a declarative language with a precisely defined software-independent semantics. XPath 2.0 and XSLT 2.0 meet these standards [13,14].

For the conversion from PDF to XML, we reached this goal except for the first step of the transformation and the final validity check. In the first step, we used `pdftohtml -xml` (http://pdftohtml.sourceforge.net) to transform the PDF to XML. Unfortunately, this program may produce non-well-formed XML and it has problems with certain diacritics. We repaired these with a `perl` script and checked for XML well-formedness with `xmllint`. This was also used in the final step to check validity with respect to the schema specified in Relax NG.

The conversion from XML back to PDF is done by an XSLT 2.0 script that produces a LaTeX source file.

Economics We calculate what can be saved using the file size reduction based on the prices for storage and bandwidth set by Amazon, http://aws.amazon.com/s3/#pricing at the time of writing (Spring 2009). The fixed costs for storing the test collection are still very modest: $ 30 per year. The variable costs are determined by the number of users and the number of documents they want to see: downloading at Amazon costs $0.17 per GB. Table 5 lists the costs per year for 3 user models and three ways of serving: the original collection (as it is now served at http://www.statengeneraaldigitaal.nl), serving the transformed PDF's (as described in this paper), or serving the gzipped XML files only. The monetary savings are two orders of magnitude.

It is hard to quantify the amount of time needed to create the transformation scripts. This depends very much on the complexity and the regularity of the documents and the use of (commercial) ETL tools [27].

Table 5 Costs per year in dollars (rounded) for downloading documents stored at Amazon (prices April 2009)

Average document sizes are used

Daily use	Original collection	Transformed collection	Transformed collection client side XML-2-PDF
10 users 10 docs each	$100	$2	$0.64
50 users 10 docs each	$500	$11	$3
100 users 10 docs each	$1,000	$21	$6

Table 6 Percentage correctly extracted structural elements

Feature	Score (%)	Comments
Topics	77.8	All recognized, but in 22.2% included too much text
Blocks	100	
Speakers	88.7	Caused by OCR errors
Paragraphs	93.5	
Header	91.5	Caused by OCR errors
Footer	92.5	Caused by OCR errors
Stage directions	73.5	

5.2 Quality of the reconstruction

Quality of the data: from PDF to XML We now evaluate the transformation from the original PDF file to XML, which was described in 4.1. Table 2 describes the quality of extracting page header, page footers, and paragraphs and determines the correct reading order of the text per page (evaluation on 1,034 pages from 15 randomly chosen proceedings files from the period 1928–1966).

For each part in the proceedings that we wanted to extract and mark up by XML tags, we scored whether the start- and end-tags were placed correctly. We evaluated two complete days (50 pages). Table 6 shows the percentage of correctly marked elements for 7 typographical features.

These are promising initial scores. The semantically important XML elements, topic and block, were all recognized. The topic description sometimes included too much text (32.2% of the topics), but they were correctly marked as topics. Most of our mistakes were caused by OCR errors which did not let our extraction rules fire. E.g. for recognizing the start of a speech element (and the title, name, and party of the speaker), we use the pattern `[title][name][(party)][:]` as in `[De heer] [Van der Spek ][(PSP)][:]`.
But sometimes this string is wrongly tokenized by the OCR as `DeheerVanderSpek(PSP):`. OCR mistake repairing, using e.g. the TICL technique [28], will improve our scores considerably.

We remark on the rather low accuracy of 67% for the reading order, presented in Table 2. This means that in 33 of the 100 pages, at least one word on the page was not placed in the correct reading order. In the vast majority of cases, a page contained just one error: a word from the footer was added to the end of a column and thus ended up in the running text. From this observation, we can estimate the reading order quality for the structural elements in Table 6. We calculated the average length of the elements in pages using all proceedings from 1997. They are as follows, topics: 8.2 pages; blocks: 1.8 page; speeches: .2 page, and paragraphs: .07 page. If we count 2 reading errors per page, then with 5 speeches on a page, we get an expected accuracy of 87% for speeches.

Quality of the look and feel: from XML to PDF Our goal was to keep the look and feel of the structure of the documents. This was achieved with great success, see Fig. 2 for an indication of the results.

We wanted to preserve the layout of each page as much as possible, but not be overly restrictive. For instance, old and new pages should contain exactly the same characters in the same order, but the words may be broken differently over the lines of texts. Table 7 contains the most important typographical features of these documents and an assessment of the quality of our reconstruction. We tested the similarity each time on ten randomly chosen documents. We score whether the element was visually indistinguishable throughout the complete document. Figure 3 contains examples of visual (in)distinguishability. In the top of the figure, we see two footers (on the left the original). These were scored as being "the same". Below that, we see twice the wording of a *motion* (the original is on the left). The header *Motie* between the two horizontal lines indicates an important structural feature of the text, which is not present in the reconstruction. For that reason, these two are scored as different.

An error analysis indicated that almost all mistakes are due to OCR errors or due to inconsistent layout in the original.

5.3 Additional advantages

Having the data in XML creates numerous analysis possibilities which are impossible to do automatically from the original PDF files. Marx [21] and Kaptein et al. [12] give a number of examples related to search, in particular focused retrieval and result aggregation. Here, we look at several possibilities that become available when making a new PDF file with LaTeX from XML input.

Table of contents The Dutch proceedings do not contain a table of contents. The table of contents acts as an agenda

Fig. 2 Similarity of layout between original (*right*) and our copy (*left*)

Table 7 Score of visual similarity on 10 typographical features

Feature	Score
Header	9/10
Footer	7/10
Individual speeches	6/7
Stage directions	
- Members present	4/4
- Start of agenda topics	6/10

The score `k/m` indicates that on `k` of the `m` documents containing these elements, all elements in that document were visually the same as in the original

of the meeting and hence is very valuable. It can be created automatically from the extracted structure and accurately reflects the order of the meeting.

Select from PDF Users can select text from the PDF file.

What you see is what you get The PDF files available at http://statengeneraaldigitaal.nl have a logic that is not obvious to average users. If opened in a PDF reader, one looks at the scanned images. But it is possible to search with Control F and that yields highlighted hits. However, the search takes place in the OCRed text that may contain errors. Thus, words that occur in the text may not be found due to OCR errors. This can be confusing for users. Also, seeing the OCR errors gives users the opportunity to broaden their search terms and still retrieve what they look for.

Wikification and hyperlinking Names of persons who speak may be hyperlinked to pages with their biographical information [26]. References to other parliamentary documents may be hyperlinked to these documents.

Back of the book index Using machine learning and keyword extraction techniques [11], useful index terms can be extracted and indicated in the running text. Creation of a back of the book index is then automatic using LaTeX's `makeindex` command

6 Related work

Our work is not about improving the quality of noisy data, the topic of the AND Workshop series, but rather on improving the interaction of users with a common representation of noisy data, namely facsimile images with hidden OCRed text. We focus on the task of searching and browsing through large collections of scanned documents. Directly relevant work is in the field of user interface design, in particular the design of search user interfaces [10].

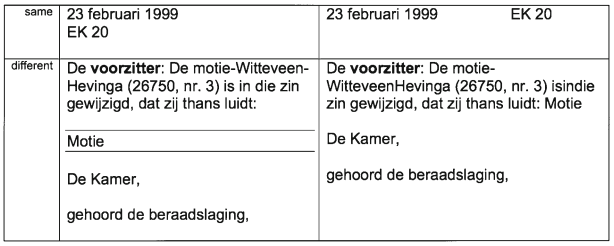


Fig. 3 Example of two features scored as the same (*top*) and two scored as different (*bottom*). At the *left* is the original, at the *right* the reconstructed document

The transformation of scanned and OCRed documents into XML is best seen as a Document Recognition and Retrieval application which does document structure analysis [19]. Here book structure recognition [8], like finding chapters and creating table of contents is closely related, as well as algorithms for deeper analysis, like extracting bibliographic data [20]. As in [15], we detect the logical structure in the document and use layout, font, and textual information. As most systems, we employ a rule-based architecture. The transformation can thus also be seen as a text extraction task (as in the TAC and MUC conferences [23,25]) combined with a document analysis task [3].

In this paper, we focus on the 'look-and-feel' of the documents. An extensive body of research on sustainable digital preservation of properties exists under the name of *significant properties* [16].

Mao et al. [28] describes the quality of the OCR of the data collection that has been used and presents a system for non-interactive error correction. This system is extensively evaluated on the Dutch Hansard dataset.

Making governmental and/or political data easily accessible through the internet is a major research area with a lot of ongoing activity. The W3C has a special interest group on eGovernment (http://www.w3.org/2007/eGov/) which encourages governments to publish their data in reusable, linkable, human- and machine-readable formats using open standards such as XML, RDF, and Dublin Core [1,2]. Independent non-profit organisations scrape governmental Web sites and create vertical search engines, mashups, or appealing visualizations, e.g. http://theyworkforyou.com and http://capitolwords.org.

Within digital curation research, XML is seen as an important data format for storing data for long periods of time. The National Archives of Australia intend to store all they have in XML and developed a software tool for this, XENA (http://xena.sourceforge.net). For the use of XML as a format for governmental documents, see [29] and the publications of the W3C eGov working group [1,2]. Another development in this direction is the UVC (Universal Virtual Computer) developed by IBM and the Dutch Royal Library [31].

7 Conclusion

We have shown that reconstructing PDF documents from scanned and OCRed data is feasible and leads to size reductions of two orders of magnitude. The quality of the reconstructed PDF files is very good and in several aspects better than the original. The most important gain of this exercise is the reduction in download time from unacceptably slow to instantaneous.

Our sample of 6 years is representative of the Dutch data from 1995 until 1878, in terms of layout complexity and noisiness of the OCR data, except for the period between the two world wars which has much lower OCR quality, due to the use of low-quality paper. Preliminary investigations abroad (Belgium, Germany, Spain) show that our findings generalize to other parliamentary proceedings corpora. Interesting directions of future research are to exploit the rich structural semantics of these documents in ways described in Sect. 5.3.

Acknowledgments Maarten Marx acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the FET-Open grant agreement FOX, number FP7-ICT-233599.

References

1. Alonso, J. et al.: Improving Access to Government Through Better Use of the Web. W3C Interest Group Note 12 May 2009. http://www.w3.org/TR/egov-improving/
2. Bennet, D., Harvey, A.: Publishing Open Government Data (W3C Working Draft 8 September 2009). http://www.w3.org/TR/gov-data/
3. Breuel, Th.: High performance document layout analysis. In: Doermann, D. (eds.) Proceedings 2003 Symposium on Document Image Understanding Technology, pp. 209–218 (2003)
4. Clarke, Ch., Agichtein, E., Dumais, S., White, R.: The influence of caption features on clickthrough patterns in web search. In: Proceedings SIGIR '07, pp. 135–142 (2007)
5. Doan, A., Ramakrishnan, R., Vaithyanathan, S.: Managing information extraction: State of the art and research directions. In: Proceedings SIGMOD '06, pp. 799–800 (2006)
6. Gielissen, T., Marx, M.: Exemelification of parliamentary debates. In: Proceedings of the 9th Dutch-Belgian Information Retrieval Workshop (DIR 2009), pp. 19–25. Twente, The Netherlands (2009)
7. Gladney, H.M., Lorie, R.A.: Trustworthy 100-year digital objects: Durable encoding for when it's too late to ask. ACM Trans. Inf. Syst. **23**(3), 299–324 (2005)
8. He, F., Ding, X.: Hierarchical logical structure extraction of book documents by analyzing table of contents. In: Proceedings of the SPIE Conference on Document Recognition and Retrieval XI, pp. 6–13 (2004)

9. Hearst, M.: Design recommendations for hierarchical faceted search interfaces. In: ACM SIGIR Workshop on Faceted Search (2006)
10. Hearst, M.: Search User Interfaces. Cambridge University Press, linebreak Cambridge (2009)
11. Hulth, A., Karlgren, J., Jonsson, A., Boström, H., Asker, L.: Automatic keyword extraction using domain knowledge. In: Proceedings CICLing 2001, pp. 472–482. Springer (2001)
12. Kaptein, R., Marx, M., Kamps, J.: Who said what to whom? Capturing the structure of debates. In: Proceedings SIGIR '09, pp. 831–832 (2009)
13. Kay, M.: XPath 2.0 Programmer's Reference. Wrox, Birmingham (2004)
14. Kay, M.: XSLT 2.0 3rd edn Programmer's Reference. Wrox, Birmingham (2004)
15. Klink, S., Dengel, A., Kieninger, T.: Document structure analysis based on layout and textual features. In: Proceedings of International Workshop on Document Analysis Systems (2000)
16. Knight, G., Pennock, M.: Data without meaning: Establishing the significant properties of digital research. In: iPRES 2008 Conference Proceedings (2008)
17. Koninklijke Bibliotheek: Staten-generaal digitaal (2009). http://www.statengeneraaldigitaal.nl/backgrounds.html
18. Ludäscher, B., Mukhopadhyay, P., Papakonstantinou, Y.: A transducer-based XML query processor. In: Proceedings VLDB '02, pp. 227–238. VLDB Endowment (2002)
19. Mao, S., Rosenfeld, A., Kanungo, T.: Document structure analysis algorithms: A literature survey. In: Proceedings of the SPIE Conference on Document Recognition and Retrieval X, pp. 197–207 (2003)
20. Mao, S., Kim, J., Thoma, G.: Style-independent document labeling: Design and performance evaluation. In: Proceedings of the SPIE Conference on Document Recognition and Retrieval XI, pp. 14–22 (2004)
21. Marx, M.: (2009) Long, often quite boring, notes of meetings. In: ESAIR '09: Proceedings of the WSDM '09 Workshop on Exploiting Semantic Annotations in Information Retrieval, pp. 46–53. ACM (2009)
22. Marx, M., Schuth, A.: DutchParl. A corpus of parliamentary documents in Dutch. In: Proceedings Language Resources and Evaluation (LREC) pp. 3670–3677 (2010)
23. Message Understanding Conference Proceedings MUC-7: National Institute of Standards and Technology (NIST) Gaithersburg, Maryland, USA (1997)
24. Murdock, V., Lalmas, M.: Workshop on aggregated search. SIGIR Forum **42**(2), 80–83 (2008)
25. Proceedings of the First Text Analysis Conference (TAC 2008): National Institute of Standards and Technology (NIST) Gaithersburg, Maryland, USA (2008)
26. Rada, M., Andras, C.: Wikify!: Linking documents to encyclopedic knowledge. In: Proceedings CIKM '07, pp. 233–242 (2007)
27. Rahm, E., Do, H.H: Data cleaning: Problems and current approaches. IEEE Tech. Bull. Data Eng. **23**(4), 3–13 (2000)
28. Reynaert, M.: Non-interactive OCR post-correction for giga-scale digitization projects. In: Proceedings of the CICLing (Computational Linguistics and Intelligent Text Processing, 9th International Conference), pp. 617–630 (2008)
29. Salminen, A.: Building digital government by XML. In: Proceedings of the Thirty-Eighth Hawaii International Conference on System Sciences. IEEE Computer Society (2005)
30. Sigurbjörnsson, B.: Focused information access using XML element retrieval. PhD thesis, University of Amsterdam (2006)
31. Van Der Hoeven, J.R., Van Diessen, R.J., Van Der Meer, K.: Development of a universal virtual computer (uvc) for long-term preservation of digital objects. J. Inf. Sci. **31**(3), 196–208 (2005)